



Building your knowledge,
making your way!

Visão

Com a crescente demanda sobre Tecnologias, percebemos que muitas pessoas apesar de buscarem informações, não possuem fontes que queiram realmente passar o conhecimento da maneira como ela deve ser, livre e com embasamento técnico que permita ser aplicado e utilizado quando necessário, além de serem testados em sua criação, tornando esta informação útil e confiável.

Missão

O Laboratório foi criado com a intenção de buscar e disseminar o conhecimento de uma maneira clara e objetiva, de forma gratuita, auxiliando na evolução dos membros e da sociedade na qual estas informações são compartilhadas, buscando o crescimento de todos os envolvidos nesta criação de valores.

Licença

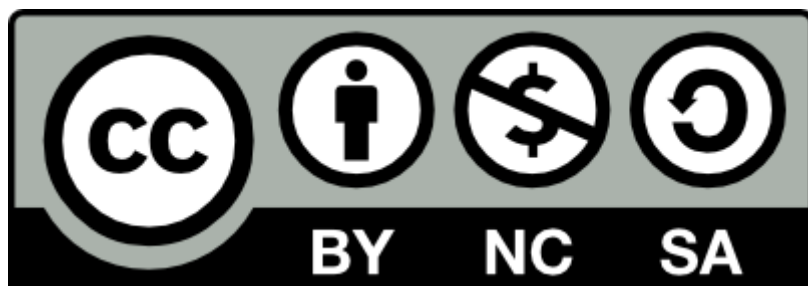


Figura 01 – Licença Criative Commons – by-nc-as

Esta licença permite que outros remixem, adapte, e criem obras derivadas sobre a obra original, desde que com fins não comerciais e contanto que atribuam crédito ao autor e licenciem as novas criações sob os mesmos parâmetros. Outros podem fazer download ou redistribuir a obra da mesma forma que na licença anterior, mas eles também podem traduzir, fazer remixes e elaborar novas histórias com base na obra original. Toda nova obra feita a partir desta deverá ser licenciada com a mesma licença, de modo que qualquer obra derivada, por natureza, não poderá ser usada para fins comerciais.

This license lets other remix, tweak, and build upon your work non-commercially, as long as they credit you and license their new creations under the identical terms.

Para maiores informações sobre o método de licenciamento acesse os seguintes sites:

Brasil:

<http://creativecommons.org.br/as-licencas/>
<http://creativecommons.org/licenses/by-nc-sa/3.0/br/>

Internacional:

<http://creativecommons.org/licenses/by-nc-sa/3.0/>
<http://creativecommons.org/licenses/by-nc-sa/3.0/legalcode>

Autor: Wellington Silva



1 – Proteção da Console

Neste artigo iremos abordar as melhores práticas para deixar a console do sistema mais segura. Para isso, veremos a importância de atribuir uma senha para **BIOS (Basic Input/Output System)**, além de desabilitar o boot através de outras mídias, exceto pelo **HD (Hard Disk)**.

Veremos também a importância de atribuir uma senha no gerenciador de boot (**LILO** ou **GRUB**). Vamos ver também a restrição do acesso a console para usuários que tem senhas nulas, travar a console após um tempo de inatividade, instalação de um aplicativo para travar o terminal, na necessidade de se ausentar da console e retiramos a reinicialização do sistema operacional através do pressionamento das teclas **Ctrl+Alt+Del** e **CTRL+SysRQ+B**.

2 – Proteção na Inicialização da Máquina

2.1 – Proteção no BIOS (Basic Input Output System)

O **BIOS** como o próprio nome diz, é um sistema básico de entrada e saída.

O **BIOS** é um software básico, necessário para a inicialização da placa-mãe, responsável por checar os dispositivos instalados, e carregar o sistema operacional, o que pode ser feito a partir do **HD, CD-ROM/DVD-ROM, floppy disk, pen-drive**, ou qualquer outra mídia disponível. O **BIOS** inclui também o **Setup**, o software que permite configurar as diversas opções oferecidas pela placa-mãe. O processador é programado para procurar e executar o **BIOS** sempre que o computador for ligado, processando-o da mesma forma que outro software qualquer.

Para armazenar as configurações feitas no **Setup** utilizamos o **CMOS (Complementary Metal-Oxide Semiconductor)**. Como as configurações do **Setup** representam um pequeno volume de informações, ele é bem pequeno em capacidade. Assim como a memória **RAM** principal, ele é volátil, de forma que as configurações são perdidas quando a alimentação elétrica é cortada. Por isso, toda placa-mãe inclui uma bateria, que mantém as configurações quando o micro é desligado.

Desta forma é importante proteger este software que é iniciado toda vez que o computador é ligado, pois ele é a porta de entrada para todos os outros softwares contidos nas mídias instaladas (como sistema operacional, controladora de discos com suporte a gerenciamento de **RAID**, entre outros).

A seguir iremos ver algumas ações que podemos tomar para proteger este software inicial.

2.1.1 – Habilitar Senha para Acessar o Setup

Antes da instalação de qualquer sistema operacional em seu computador, configure uma senha para acessar o **Setup** e habilite o **Boot** a partir do **CD-ROM** (Caso você deseje instalar o sistema operacional através de um **CD-ROM**). Após a instalação você deve configurar o **Setup**, e alterar a sequência de inicialização, desabilitando o **Boot** a partir de outras mídias, que não seja a do sistema operacional. Se você não fizer assim,

Autor: Wellington Silva



um cracker só precisará de acesso físico ao computador e dar o **Boot** através de outro dispositivo para acessar o sistema inteiro.

Observação: Cada fabricante, e até mesmo **BIOS** do mesmo fabricante, muda de máquina para máquina, acrescentando mais recursos ou retirando opções que não são mais usadas. Aqui utilizaremos a **BIOS** do **VMWare Workstation®**.

Para entrar nas configurações (**Setup**) basta pressionar a tecla **F2** assim que a máquina for ligada.



Figura 02 – Acesso ao Setup

Após acessar a **Setup**, vamos navegando, com as teclas de navegação do teclado até a guia **Security** e clicamos em **Set Supervisor Password**, atribuímos uma senha e confirmamos logo em seguida.

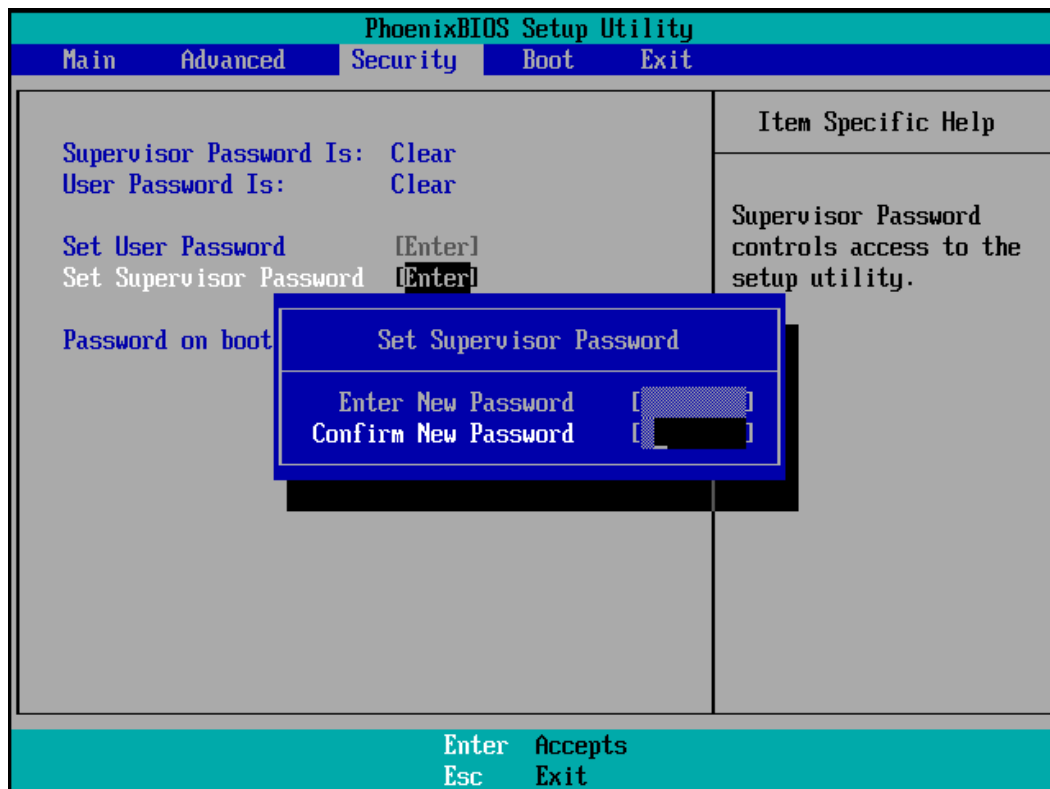


Figura 03 – Atribuindo uma Senha para Acessar o Setup

Agora vamos para a guia **Boot**, selecionamos o dispositivo de **Hard Drive** e clicamos na tecla **+** até que **Hard Drive** esteja na primeira posição.

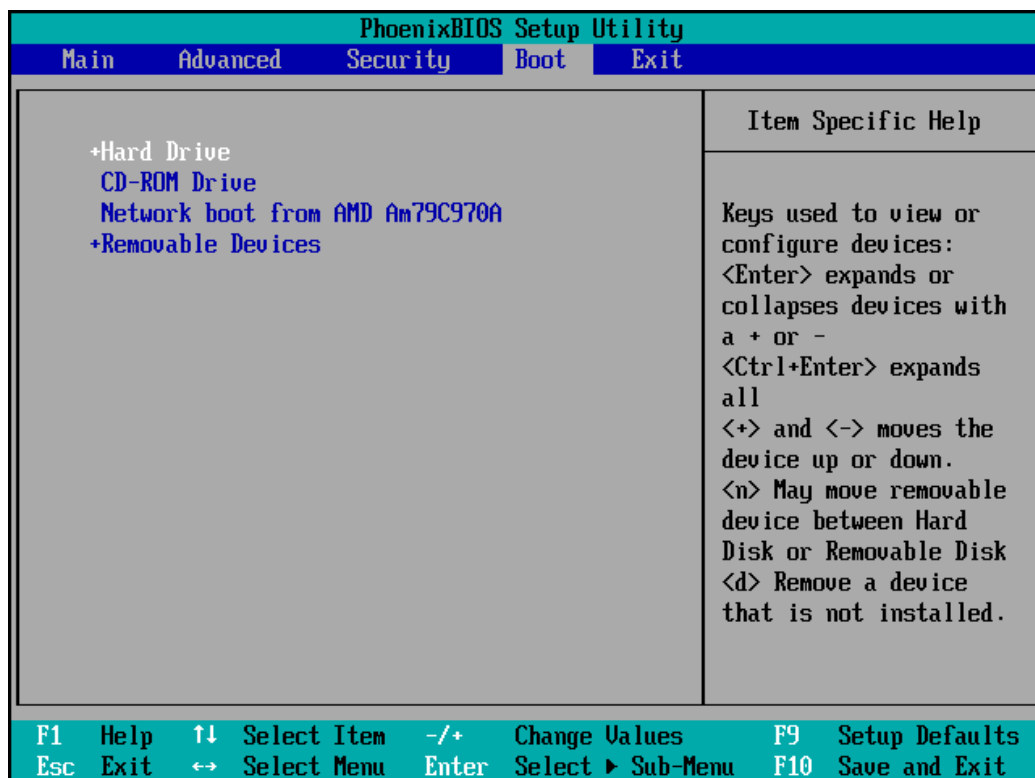


Figura 04 – Priorizando a Inicialização do Sistema Operacional pelo HD

Autor: Wellington Silva



Agora vamos até a guia **Exit** selecione a opção **Exit Saving Changes**, na caixa de diálogo **Setup Configuration**, confirme clicando em **Yes**.

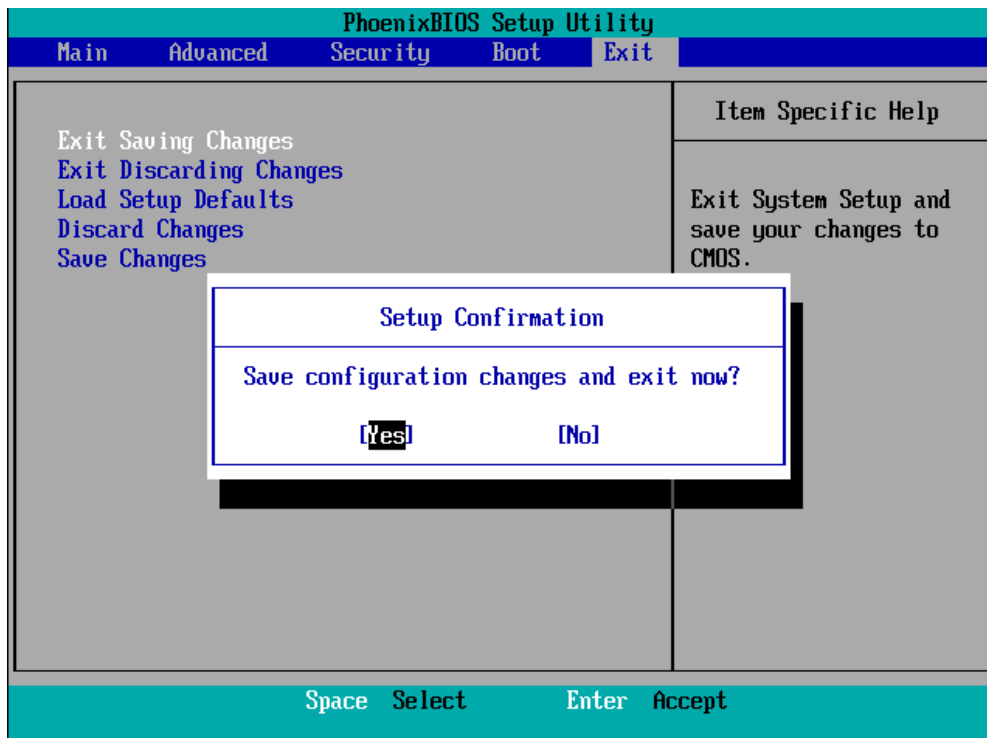


Figura 05 – Saindo e Salvando as Configurações no Setup

Agora, basta testarmos o acesso e verificar se realmente ele nos solicitará a senha na tentativa de acesso ao **Setup**.

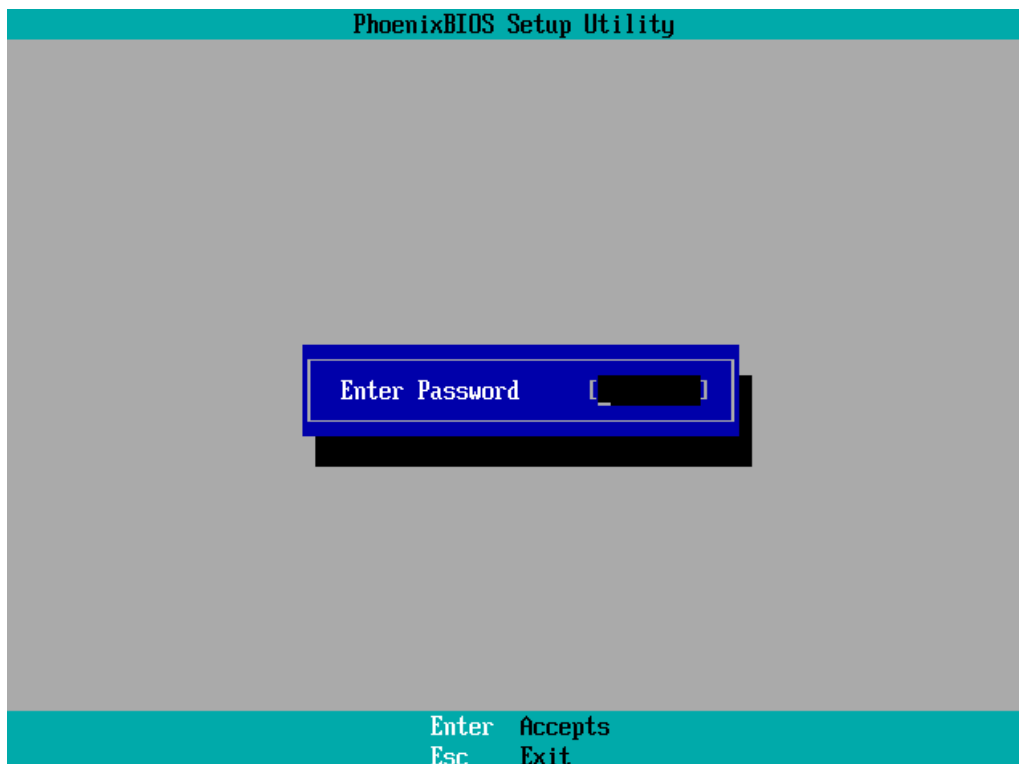


Figura 06 – Testando o Acesso ao Setup

Autor: Wellington Silva



Observação: Tome cuidado com a opção **Password on Boot** na guia **Security**, pois se habilitarmos esta opção ele pedirá a senha sempre que a máquina for iniciada. Sendo que se a máquina for reiniciada por algum motivo o sistema operacional não será carregado enquanto alguém não colocar a senha para continuar o processo de **Boot**.

Nota: Em casos muito específico, esta opção de senha na inicialização pode ser usada. Um exemplo, é em sistemas que não pode, de forma nenhuma ser reiniciado sem autorização de uma chefia, com isso, garantimos que somente pessoas autorizadas e que tenha acesso físico possam reiniciar a máquina.

Outra possibilidade para proteger tanto o sistema operacional como as informações contidas no **HD** pode ser usar criptografia total do disco, porém temos que utilizar esta opção com muito cuidado, pois na perda do certificado, todas as informações serão perdidas, sem contar na impossibilidade de aplicar técnicas de **forensics post-mortem**.

2.2 – Proteção nos Gerenciadores de Boot

Gerenciadores de **Boot** são softwares capazes de iniciar o processo de carregamento do(s) sistema(s) operacional(is) em um computador. São de extrema importância em casos de computadores com mais de um sistema operacional instalado, pois os gerenciadores tem a função de permitir ao usuário a escolha do sistema a ser carregado.

Os principais gerenciadores de **Boot** existentes são: **LILO (Linux Loader)** e **GRUB (GRand Unified Bootloader)** para sistemas **GNU/Linux**, **NTLDR (New Technology Loader)** para a família **MS Windows** e **BTX Loader** para família **BSD**.

Aqui devemos proteger a edição do nosso gerenciador de Boot para que uma pessoa com acesso a console não possa alterar a senha de **root** ou de qualquer outro usuário, passando como parâmetro para o Kernel o argumento **"init=/bin/sh rw"**.

Para assegurarmos que essa técnica não seja usada contra nossos sistemas, você deverá definir uma senha para o gerenciador de Boot.

2.2.1 – Protegendo o Gerenciador de Boot LILO

O **LILO (Linux LOader)** é o mais antigo, o mais conhecido e distribuído gerenciador de boot para o **Linux**. Ele é dividido em duas partes. A primeira parte é um pequeno código residente no setor de boot ou na **MBR (Master Boot Record)**. Este pequeno código funciona como um apontador que procura pela segunda parte do código que pode estar em outro lugar no disco.

Esta segunda parte é bem mais complexa. Ela oferece ao usuário uma linha de comando (prompt) para que ele escolha qual imagem do Kernel deverá ser carregada para a memória e fornecer parâmetros adicionais para o sistema e iniciá-lo.

Esta divisão em dois estágios se faz necessária porque o setor de Boot dos discos é muito pequeno para armazenar todo o gerenciador de Boot.

A seleção de qual imagem do Kernel, sistema operacional e a passagem de parâmetros ao Kernel poderão ser feitas automaticamente, ou usando o aviso de Boot do **LILO**.

Pressionando a tecla **TAB**, na linha de comando do **LILO**, poderá ser visto a lista de imagens de Kernels disponíveis.

Autor: Wellington Silva



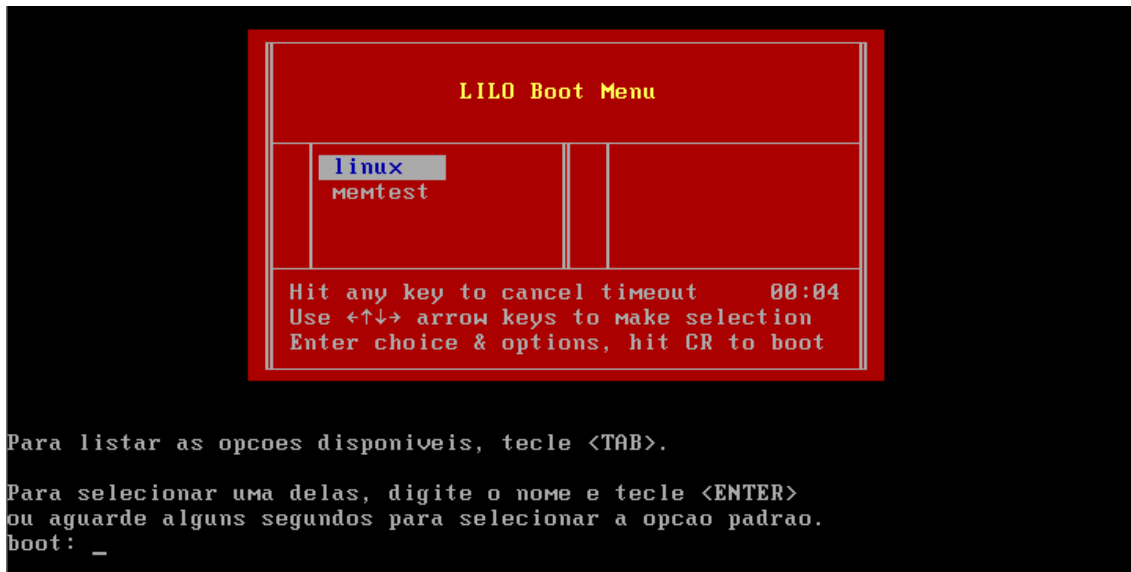


Figura 07 – Gerenciador de Boot LILO

O **LILO** possui um arquivo de configuração chamado **/etc/lilo.conf** que contém a configuração da localização das imagens do Kernel, configurações de vídeo, imagem de Kernel padrão, entre outros parâmetros.

O **LILO** utiliza um programa chamado "instalador de mapas", **lilo**, que lê as configurações do arquivo **/etc/lilo.conf** e grava em um arquivo de mapas. Este arquivo é utilizado pelo gerenciador de boot para localizar e carregar a imagem do Kernel ou outros sistemas operacionais.

Toda vez que uma alteração no Kernel for realizada ou uma nova versão for instalada, é necessário configurar o arquivo **lilo.conf** e executar o utilitário **lilo**.

Para proteger o **LILO** nós precisamos editar o arquivo de configuração **/etc/lilo.conf** e adicionar duas linhas no arquivo. As linhas a ser adicionada são **password** e **restricted** como o exemplo abaixo:

```
root@fusion:~# vi /etc/lilo.conf
boot=/dev/sda
map=/boot/map
install=/boot/boot.b
prompt
timeout=50
linear
message=/boot/message
image=/boot/vmlinuz-2.4.5-9cl
    label=linux
    root=/dev/sda1
    initrd=/boot/initrd-2.4.5-9cl.img
    read-only
    password=N@0T3Cont0 <-----+ Adicionar estas
    restricted <-----+ linhas
image=/boot/memtest86
    label=memtest

root@fusion:~#
```

Quando terminar, execute o comando **lilo** para recompilar o **lilo** no **/boot**.

Autor: Wellington Silva




```
root@fusion:~# lilo
Added linux *
Added memtest

root@fusion:~#
```

Pronto, porém, antes de testarmos vamos reforçar a proteção do arquivo **/etc/lilo.conf** atribuindo as permissões para **600 (rw-----)** e garantindo que o dono e o grupo seja o **root**, já que o **LILLO** mantém a senha em cleartext (texto puro)

```
root@fusion:~# ls -la /etc/lilo.conf
-rw-r--r--  1 root    root      271 Jul 24 01:42 /etc/lilo.conf

root@fusion:~# chmod 600 /etc/lilo.conf

root@fusion:~# chown root.root /etc/lilo.conf

root@fusion:~# ls -la /etc/lilo.conf
-rw-----  1 root    root      271 Jul 24 01:42 /etc/lilo.conf

root@fusion:~#
```

Agora reiniciamos a máquina e vamos tentar passar algum parâmetro para o Kernel.

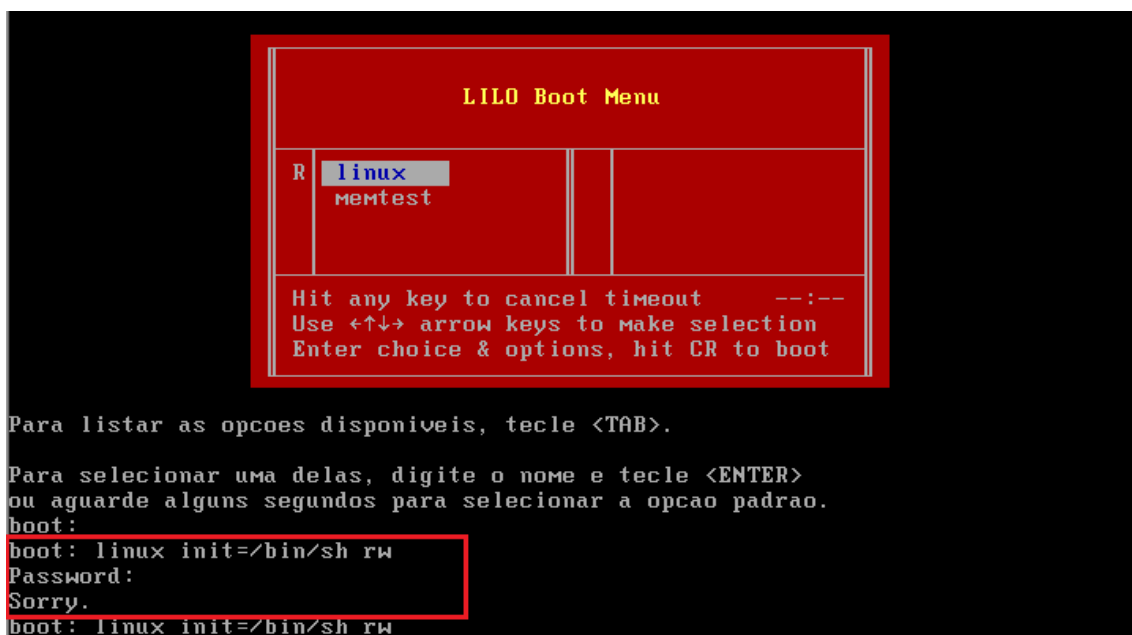


Figura 08 – Gerenciador de Boot LILLO Protegido com Senha

Apesar das distros mais recentes usarem o **GRUB** como gerenciador de boot padrão, é importante sabermos como proteger o **LILLO**, já que muitos sistemas Linux Legados ainda utilizam este gerenciador de boot. Como exemplo, alguns **PABX** ainda utilizam este gerenciador.

Outra medida de segurança importante é modificar o atributo do arquivo para imutável, evitando que o arquivo seja alterado, apagado ou que links simbólicos sejam criados, o que reduz a possibilidade de erros acidentais e alterações não autorizadas.

```
root@fusion:~# lsattr /etc/lilo.conf
----- /etc/lilo.conf

root@fusion:~# chatter +i /etc/lilo.conf
```

Autor: Wellington Silva



```
root@fusion:~# lsattr /etc/lilo.conf
----i----- /etc/lilo.conf

root@fusion:~#
```

2.2.2 – Protegendo o Gerenciador de Boot GRUB

GRUB (GRand Unifield Bootloader) é um gerenciador de boot desenvolvido inicialmente por *Erich Stefan Boleyn*, disponibilizado como software **GNU**. Entre seus principais recursos está a capacidade de trabalhar com diversos sistemas operacionais, como o Linux, o Windows e as versões BSD e consequentemente o suporte a vários sistemas de arquivos, como o **EXT2/3/4**, **ReiserFS**, **FAT**, **FFS**, entre outros.

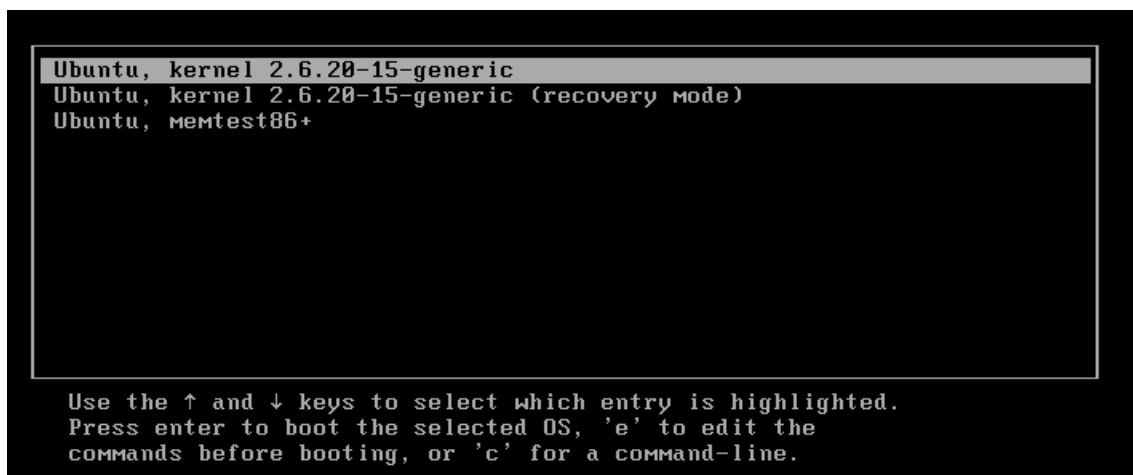


Figura 09 – GRUB v1 no Ubuntu

Um dos motivos mais óbvios para o **GRUB** ser usado é sua capacidade de permitir que o usuário escolha um dos sistemas operacionais instalados em seu computador. Em outras palavras, o **GRUB** é capaz de trabalhar com "**multiboot**". Além disso, esse gerenciador também é capaz de "**bootar**" sistemas em discos **SCSI** ou mesmo carregá-los através de imagens disponíveis em rede.

Caso esteja utilizando o **GRUB** como gerenciador de boot (eu recomendo o uso do **GRUB**), edite o arquivo **/boot/grub/menu.lst** nas versões mais antigas (**GRUB v1**).

No **GRUB Legacy (primeiras versões do GRUB que atualmente não está mais em desenvolvimento)** o procedimento é semelhante à configuração do **LILO**, onde precisamos acrescentar as seguintes linhas no topo do arquivo. Isto evita que usuários editem os itens de inicialização.

```
root@fusion:~# vi /boot/grub/menu.lst
timeout = 15
password N@T3C0nt0

===[ Resumido ]==
```

Pronto, porém, antes de testarmos vamos reforçar a proteção do arquivo **/boot/grub/menu.lst** atribuindo as permissões para **600 (rw-----)** e garantindo que o dono e o grupo seja o **root**, já que o **LILO** mantém a senha em cleartext (texto puro)

Autor: Wellington Silva



```

root@fusion:~# ls -la /boot/grub/menu.lst
-rw-r--r--  1 root    root      271 Jul 24 01:42 /boot/grub/menu.lst

root@fusion:~# chmod 600 /boot/grub/menu.lst

root@fusion:~# chown root.root /boot/grub/menu.lst

root@fusion:~# ls -la /boot/grub/menu.lst
-rw-----  1 root    root      271 Jul 24 01:42 /boot/grub/menu.lst

root@fusion:~#

```

Agora vamos reiniciar a máquina e validar se a edição do **GRUB** está protegida pela senha atribuída no arquivo **menu.lst**.

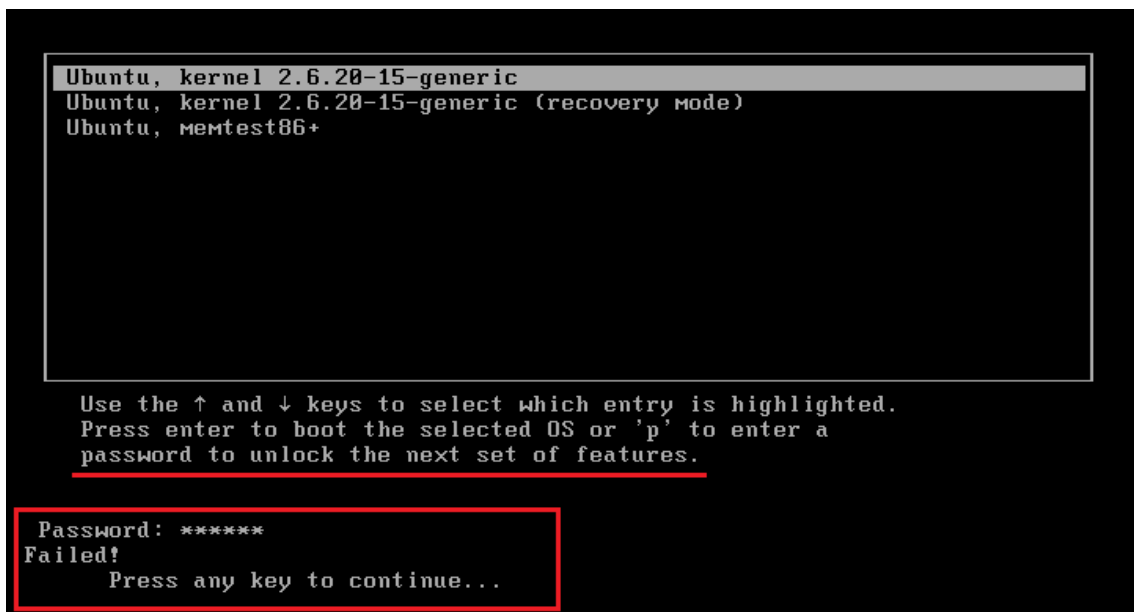


Figura 10 – GRUB legacy Protegido por Senha

Para não deixar a senha em texto puro podemos ainda inserir a senha já cifrada com o utilitário **grub-md5-crypt**, assim a senha fica mais protegida contra olhares aguçados.

No **prompt** digite o comando **grub-md5-crypt**, digite a senha e confirme.

Após isso, ele mostrará a senha digitada no formato **MD5**. Veja no exemplo a seguinte:

```

root@fusion:~# grub-md5-crypt
Password:
Retype password:
$1$IX94l0$yaIS0WVn7GAdTlo/Tamrk.

root@fusion:~#

```

Para especificar no **GRUB** que uma senha está no formato **MD5** use a seguinte diretiva:

```

root@fusion:~# vi /boot/grub/menu.lst

timeout      3
password --md5 $1$IX94l0$yaIS0WVn7GAdTlo/Tamrk.

--=[ Resumido ]==

```

Autor: Wellington Silva



O parâmetro **--md5** foi adicionado para instruir o **grub** a fazer o processo de autenticação **MD5**. A senha fornecida é a **N@0T3C0nt0** com encriptação **MD5**. O uso do método de senhas **MD5** é recomendado em contrapartida ao uso de senhas em texto plano.

Agora no **GRUB 2** o processo é um pouco diferente. E agora é necessário criar um usuário além da senha e a senha pode ser gravada em **SHA512** que é mais segura que o **MD5** do **GRUB Legacy**.

Para colocarmos senha no **GRUB** devemos agora editar um arquivo de cabeçalho e recompilar o **GRUB** para que ele acrescente as linhas no novo arquivo de configuração principal, que se encontra em **/boot/grub/grub.cfg**. O arquivo de cabeçalho que iremos acrescentar o recurso de senha é o **/etc/grub.d/00_header**.

Adicione no final do arquivo o seguinte:

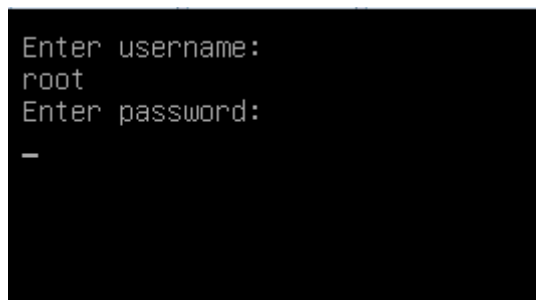
```
cat << EOF
set superusers="root"
password root N@0T3C0nt0
EOF
```

Com isso, criamos um usuário chamado **"root"** com a senha **"N@0T3C0nt0"**

Para concluir vamos recompilar o **GRUB** para que ele gere o novo arquivo **grub.cfg**:

```
root@fusion:~# update-grub
Generating grub.cfg ...
Found linux image: /boot/vmlinuz-2.6.32-5-686
Found initrd image: /boot/initrd.img-2.6.32-5-686
done
root@fusion:~#
```

Pronto! Vamos testar, reinicie a máquina e na tela do **GRUB** clique em **"e"** (de **Edit**) e verifique que ele solicitará a senha de acesso.



```
Enter username:
root
Enter password:
_
```

Figura 11 – GRUB com Senha

Colocando a senha correta podemos editar os parâmetros de Kernel, como pode ser visto abaixo:

```

GNU GRUB  version 1.98+20100804-14+squeeze1

insmod part_msdos
insmod ext2
set root='(hd0,msdos1)'
search --no-floppy --fs-uuid --set 30760ba9-dc35-4e83-b286-5fa9ba379\
9a6
echo 'Loading Linux 2.6.32-5-686 ...'
linux /vmlinuz-2.6.32-5-686 root=UUID=40ef5be0-9a5e-44c7-8114-d0cc55\
3df50a ro quiet
echo 'Loading initial ramdisk ...'
initrd /initrd.img-2.6.32-5-686

Minimum Emacs-like screen editing is supported. TAB lists
completions. Press Ctrl-x to boot, Ctrl-c for a
command-line or ESC to discard edits and return to the
GRUB menu.

```

Figura 12 – GRUB Sendo Editado Após a Validação de Usuário e Senha

Para não deixarmos a senha em texto puro, podemos ainda inserir, no arquivo **/etc/grub.d/00_header**, a senha já cifrada com o utilitário **grub-mkpassword-pbkdf2**, assim a senha fica mais protegida contra olhares aguçados.

Vamos à prática:

```

root@fusion:~# grub-mkpasswd-pbkdf2
Enter password:
Reenter password:
Your PBKDF2 is
grub.pbkdf2.sha512.10000.4EFC05BB722F3CBA65A26C5F0D8457E90ACFE8CBD72E5E5FE434
6BAAA23BD8B1C261434BDBD4D8F635A4E1A9B8117150FFEA38441D605A4F4040F3D25D0C698D.
25BBA9EEEE5F38134BE5E339C5AA00C28FC4E5F7021377B5ADDCED7EE6A51FA60C65647BCC2BE3
4CCA970F03D185FAF619A8795DAEB94DE894294AB8C332C99D4

root@fusion:~#

```

Essa é a nossa senha em **hash SHA512** agora podemos substituir no arquivo **/etc/grub.d/00_header** ou substituir direto no arquivo de configuração do **GRUB** em **/boot/grub/grub.cfg** (desta forma não precisamos recompilar o **GRUB** - poderíamos ter feito direto no arquivo o exemplo anterior, mas para efeitos didáticos usamos a forma correta, que é recompilando o **GRUB**).

Para nosso exemplo entramos no arquivo de configuração do **GRUB**, deletamos a linha onde tinha a senha em texto puro e no lugar colocamos o **hash** gerado pelo **grub-mkpasswd-pbkdf2**.

```

root@fusion:~# vi /boot/grub/grub.cfg

--=[ Resumido ]==

set timeout=5
set superusers="root"

```

Autor: Wellington Silva



```
password_pbkdf2 root grub.pbkdf2.sha512.10000.4EFC05BB722F3CBA65A26C5F0D8457E
90ACFE8CBD72E5E5FE4346BAAA23BD8B1C261434BDBD4D8F635A4E1A9B8117150FFEA38441D60
5A4F4040F3D25D0C698D.25BBA9EEE5F38134BE5E339C5AA00C28FC4E5F7021377B5ADDCED7EE
6A51FA60C65647BCC2BE34CCA970F03D185FAF619A8795DAEB94DE894294AB8C332C99D4
```

Observação: Caso a senha em **Hash SHA512** for definida através do arquivo **/etc/grub.d/00_header**, devemos especificar **password_pbkdf2** ao invés de **password**.

Outra medida de segurança importante é modificar o atributo do arquivo para imutável, evitando que o arquivo seja alterado, apagado ou que links simbólicos sejam criados, o que reduz a possibilidade de erros acidentais e alterações não autorizadas.

```
root@fusion:~# lsattr /etc/grub.d/00_header
----- /etc/grub.d/00_header

root@fusion:~# chattr +i //etc/grub.d/00_header

root@fusion:~# lsattr /etc/grub.d/00_header
----i----- /etc/grub.d/00_header

root@fusion:~#
```

3 – Proteção dos Terminais na Console

Quando falamos em segurança, a primeira coisa que vem à mente é um possível ataque remoto. Então, os nossos servidores que não estão conectados à Internet não estão correndo nenhum perigo, certo? Errado, aliás, dois fatores estão errados. Não está conectado diretamente à Internet, não quer dizer que está totalmente seguro, pois, se um cracker conseguir passar pelo nosso firewall, ele vai conseguir acesso às outras máquinas que estão conectadas à rede interna e não necessariamente conectada diretamente à Internet.

Outro fator é funcionários mal-intencionado, com acesso local (físico) aos nossos servidores. Muitas vezes quem entende de tecnologia costuma subestimar as pessoas com cargos mais braçais (como funcionário que trabalham no chão de fábrica, pessoal da faxina, motoristas, etc). Ledo engano, pois esse pessoal tem computadores em suas casas e mesmo sem intenções maliciosas podem causar problemas ao ver um terminal em sua frente. Além disso, se por um descuido da equipe de TI, um funcionário mal-intencionado conseguir acesso ao Datacenter, se passando por não entender nada de tecnologia, pode na verdade ser um espião industrial com conhecimentos superior aos seus!

3.1 – Removendo Shells Desnecessárias

O **Shell** no Linux tem a função de mediador entre o Kernel e o usuário. Interpretando os comandos executados na console. O acesso ao Shell deve ser restrito a usuários autorizados que fará login no sistema.

Vamos remover as Shell válidas de todos os usuários que não vão executar oficialmente login no sistema, por meio de um terminal local (**tty**) ou via **SSH**. Isso garante que se uma aplicação for executada por um usuário como **www-data**, que é usado para subir o serviço de WEB Server, se for comprometido, esse usuário não poderá efetuar login no sistema e nem utilizar indevidamente o sistema podendo comprometer através da criação ou instalação de backdoors, rootkits e ataques de buffer overflow.

Combinado a isso, crie um usuário que utilizaremos para administrar o servidor e atribua uma senha para esse usuário.

```
root@fusion:~# useradd -m -s /bin/bash W3ll
root@fusion:~# passwd W3ll
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
root@fusion:~#
```

Vamos olhar o arquivo que guarda as informações de login, para vermos quais shell estão atribuídas aos usuários.

```
root@fusion:~# cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin:/bin/sh
sys:x:3:3:sys:/dev:/bin/sh
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/bin/sh
man:x:6:12:man:/var/cache/man:/bin/sh
lp:x:7:7:lp:/var/spool/lpd:/bin/sh
mail:x:8:8:mail:/var/mail:/bin/sh
```

Autor: Wellington Silva



```
news:x:9:9:news:/var/spool/news:/bin/sh
uucp:x:10:10:uucp:/var/spool/uucp:/bin/sh
proxy:x:13:13:proxy:/bin:/bin/sh
www-data:x:33:33:www-data:/var/www:/bin/sh
backup:x:34:34:backup:/var/backups:/bin/sh
list:x:38:38:Mailing List Manager:/var/list:/bin/sh
irc:x:39:39:ircd:/var/run/ircd:/bin/sh
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/bin/sh
nobody:x:65534:65534:nobody:/nonexistent:/bin/sh
libuuid:x:100:101::/var/lib/libuuid:/bin/sh
w3ll:x:1000:1000:w3ll,,,:/home/w3ll:/bin/bash
sshd:x:101:65534::/var/run/sshd:/usr/sbin/nologin
W3ll:x:1001:1001::/home/W3ll:/bin/bash

root@fusion:~#
```

Observem a quantidade de usuários que não necessitam de login, nem local muito menos remoto e que tem shell válida. Isso se torna um problema para a segurança quando temos um comprometimento dos serviços que usam usuários do sistema para sua gerencia.

Por isso, vamos criar um script para remover todas as shell válida dos usuários, com exceção daqueles que for necessário o acesso (como a conta de uso da equipe de TI e do root).

```
root@fusion:~# > /usr/local/sbin/noshell

root@fusion:~# chmod 700 /usr/local/sbin/noshell

root@fusion:~# vi /usr/local/sbin/noshell

#!/bin/bash

## CORES
amarelo="\e[33;1m"
azul="\e[34;1m"
verde="\e[32;1m"
vermelho="\e[31;1m"
fim="\e[m"

AUDITPATH=$1

if [ -z "$1" ] || [ "$1" == " " ]; then
    echo -e "$vermelho ERRO: POR FAVOR, INDICAR O CAMINHO E O NOME DE UM
ARQUIVO"
    echo -e " ONDE SERA GRAVADO UMA COPIA PARA POSTERIOR AUDITORIA.\012"
    echo -e " ====="
    echo -e " = EXEPLO = "
    echo -e " ====="
    echo -e " # noshell /root/auditoria/passwd.audit$fim"
    exit 1
fi

if [ `whoami` != "root" ]; then
    echo -e "$vermelho EXECUTE SOMENTE COMO ROOT$fim"
    exit 1
fi
```

Autor: Wellington Silva




```

rel()
{
    echo -e "$amarelo GERANDO RELATORIO... $fim"
    echo -e "\011\011Auditoria feita em: `date`" >> $AUDITPATH
    cat /etc/passwd >> $AUDITPATH
    echo -e "\012\012" >> $AUDITPATH
    chown root.root $AUDITPATH
    chmod 600 $AUDITPATH
}
rel

for USERSHELL in `cat /etc/passwd | grep -v false | cut -f1 -d":" | grep -v root` ;do
    echo -e "DESEJA $vermelho REMOVER$fim A SHELL DO USUARIO $vermelho $USERSHELL$fim? (S -> SIM): "
    read ACAO

    case $CAOA in
        S|s)
            usermod -s /bin/false $USERSHELL
            echo -e "$verde SHELL REMOVIDA COM SUCESSO$fim"
            ;;
        *)
            echo -e "$verde SHELL MANTIDA$fim"
            ;;
    esac
done

exit 0

```

Neste script que criamos, iremos fazer um backup do arquivo original no caminho passado após o script (por exemplo, **noshell /root/auditoria/passwd.audit**). Em seguida ele garante que o dono do arquivo de auditoria seja o **root** e que somente ele tenha permissão de ler e alterar.

Depois iremos fazer um laço (usando o **for**) para garantir que ele leia o arquivo **/etc/passwd** excluindo da consulta os usuários que não tem shell (shell igual a **/bin/false**), depois recortando somente o nome dos usuários e excluindo da pesquisa o **root** (que não pode ficar sem shell). Lembre-se de não retirar a **shell** do usuário que criamos anteriormente.

Por fim, o script mostrara cada usuário e perguntará se desejamos retirar a shell dos mesmos, digitamos **S** para que a **shell** seja removida e qualquer outra tecla para que não seja removido a shell.

Vamos à prática:

```

root@fusion:~# noshell
ERRO: POR FAVOR, INDICAR O CAMINHO E O NOME DE UM ARQUIVO
ONDE SERA GRAVADO UMA COPIA PARA POSTERIOR AUDITORIA.

=====
= EXEPL0 =
=====
# noshell /root/auditoria/passwd.audit
root@fusion:~# noshell /root/auditoria/passwd.audit
GERANDO RELATORIO...
DESEJA REMOVER O SHELL DO USUARIO sys? (S -> SIM):
s

```

Autor: Wellington Silva



```

SHELL REMOVIDA COM SUCESSO
DESEJA  REMOVE A SHELL DO USUARIO  sync? (S -> SIM):
s
SHELL REMOVIDA COM SUCESSO
DESEJA  REMOVE A SHELL DO USUARIO  mail? (S -> SIM):
s
SHELL REMOVIDA COM SUCESSO
DESEJA  REMOVE A SHELL DO USUARIO  news? (S -> SIM):
s
SHELL REMOVIDA COM SUCESSO
DESEJA  REMOVE A SHELL DO USUARIO  uucp? (S -> SIM):
s
SHELL REMOVIDA COM SUCESSO
DESEJA  REMOVE A SHELL DO USUARIO  proxy? (S -> SIM):
s
SHELL REMOVIDA COM SUCESSO
DESEJA  REMOVE A SHELL DO USUARIO  www-data? (S -> SIM):
s
SHELL REMOVIDA COM SUCESSO
DESEJA  REMOVE A SHELL DO USUARIO  backup? (S -> SIM):
s
SHELL REMOVIDA COM SUCESSO
DESEJA  REMOVE A SHELL DO USUARIO  list? (S -> SIM):
s
SHELL REMOVIDA COM SUCESSO
DESEJA  REMOVE A SHELL DO USUARIO  irc? (S -> SIM):
s
SHELL REMOVIDA COM SUCESSO
DESEJA  REMOVE A SHELL DO USUARIO  gnats? (S -> SIM):
s
SHELL REMOVIDA COM SUCESSO
DESEJA  REMOVE A SHELL DO USUARIO  nobody? (S -> SIM):
s
SHELL REMOVIDA COM SUCESSO
DESEJA  REMOVE A SHELL DO USUARIO  libuuid? (S -> SIM):
s
SHELL REMOVIDA COM SUCESSO
DESEJA  REMOVE A SHELL DO USUARIO  w3ll? (S -> SIM):
n
SHELL MANTIDA
DESEJA  REMOVE A SHELL DO USUARIO  sshd? (S -> SIM):
s
SHELL REMOVIDA COM SUCESSO
DESEJA  REMOVE A SHELL DO USUARIO  w3ll? (S -> SIM):
n
SHELL MANTIDA

root@fusion:~#

```

Vamos reexecutar para vermos que ele não mostrará usuários que já passaram pela auditoria do **noshell**:

```

root@fusion:~# noshell /root/auditoria/passwd.audit
GERANDO RELATORIO...
DESEJA  REMOVE A SHELL DO USUARIO  w3ll? (S -> SIM):
n
SHELL MANTIDA

```

Autor: Wellington Silva



```
DESEJA REMOVER A SHELL DO USUARIO w311? (S -> SIM):
n
SHELL MANTIDA
root@fusion:~#
```

Pronto como vimos ele oculta todos os usuários que já tiveram suas shell removida.

Com isso todos as shell, exceto dos usuários indicados e do root, foram retiradas como podemos ver abaixo:

```
root@fusion:~# getent passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/false
bin:x:2:2:bin:/bin:/bin/false
sys:x:3:3:sys:/dev:/bin/false
sync:x:4:65534:sync:/bin:/bin/false
games:x:5:60:games:/usr/games:/bin/false
man:x:6:12:man:/var/cache/man:/bin/false
lp:x:7:7:lp:/var/spool/lpd:/bin/false
mail:x:8:8:mail:/var/mail:/bin/false
news:x:9:9:news:/var/spool/news:/bin/false
uucp:x:10:10:uucp:/var/spool/uucp:/bin/false
proxy:x:13:13:proxy:/bin:/bin/false
www-data:x:33:33:www-data:/var/www:/bin/false
backup:x:34:34:backup:/var/backups:/bin/false
list:x:38:38:Mailing List Manager:/var/list:/bin/false
irc:x:39:39:ircd:/var/run/ircd:/bin/false
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/bin/false
nobody:x:65534:65534:nobody:/nonexistent:/bin/false
libuuid:x:100:101::/var/lib/libuuid:/bin/false
w311:x:1000:1000:w311,,:/home/w311:/bin/bash <---+---- Com Shell
sshd:x:101:65534::/var/run/sshd:/bin/false /
w311:x:1001:1001::/home/w311:/bin/bash <-----'
root@fusion:~#
```

Ótimo, todos os usuários que não tinha necessidade de Shell em nosso servidor não possuem mais.

Podemos adotar como política que as próximas contas criadas com o utilitário **adduser** tenham sempre uma Shell inválida. Caso o usuário precise de uma Shell válida, podemos atribuir uma nova Shell para ele com o comando **usermod -s /bin/shell** ou editando o arquivo **/etc/passwd**.

Vamos editar o arquivo **/etc/adduser.conf** para adicionar usuários com Shell inválida.

```
root@fusion:~# vi /etc/adduser.conf

# The DSHELL variable specifies the default login shell on your
# system.
#DSHELL=/bin/bash
DSHELL=/bin/false

---[ Resumido ]---

# If DIR_MODE is set, directories will be created with the specified
# mode. Otherwise the default mode 0755 will be used.
#DIR_MODE=0755
DIR_MODE=0750
```

Autor: Wellington Silva



Nota: No **Red Hat** esses parâmetros estão no arquivo **/etc/default/useradd**, e está como **SHELL=/bin/bash**.

Além de alterarmos a Shell padrão para os novos usuários, também é interessante definirmos o permissionamento do diretório **home** dos usuários. No exemplo acima, alteramos o parâmetro **DIR_MODE** de **0755** para **0750**, desta forma, todos os usuários criados terão a permissão **0750** no seu diretório pessoal, evitando que outros usuários acessem o seu diretório.

Nota: As configurações no arquivo **/etc/adduser.conf** apenas afetam as opções padrão do comando **adduser**, para usuários criados com o comando **useradd**, deve ser especificado a Shell, através da opção **-s** (como em **-s /bin/false**) e alterar o permissionamento manualmente (algo como **chmod 600 /home/user**).

Outra forma de administrarmos a shell dos usuários é através dos utilitários **chsh** da seguinte forma:

```
root@fusion:~# echo "/bin/false" >> /etc/shells
root@fusion:~# chsh -s /bin/false www-data
root@fusion:~#
```

3.2 – Acesso aos Terminais da Console

Por questões de segurança não é interessante deixarmos o **login** habilitado em todos os terminais de texto. Por isso, vamos desabilitar alguns terminais, deixando somente 2 disponíveis. Você poderá escolher quais terminais você deseja bloquear, no nosso exemplo iremos desabilitar os terminais **tty2**, **tty3**, **tty4** e **tty5**, permitindo o **login** apenas nos terminais **tty1** e **tty6**.

Vamos editar o arquivo **/etc/inittab** comentando o uso dos terminais que iremos bloquear nas seguintes linhas:

```
root@fusion:~# vi /etc/inittab

# Note that on most Debian systems tty7 is used by the X Window System,
# so if you want to add more getty's go ahead but skip tty7 if you run X.
#
1:2345:respawn:/sbin/getty 38400 tty1
#2:23:respawn:/sbin/getty 38400 tty2
#3:23:respawn:/sbin/getty 38400 tty3
#4:23:respawn:/sbin/getty 38400 tty4
#5:23:respawn:/sbin/getty 38400 tty5
6:23:respawn:/sbin/getty 38400 tty6
```

Desta forma somente os terminais **1** e **6** poderão ser usados para login com qualquer usuário que tenha permissão de login local.

Após a edição do arquivo **/etc/inittab** devemos executar o comando **init q** para que suas alterações entrem em vigor sem a necessidade de um **restart** do sistema.

3.3 – Reboot

Caso seu sistema tenha teclado conectado, por default, qualquer um poderá reinicializar o sistema sem efetuar login. Por medidas de segurança é bom ter o controle dos acessos, inclusive de quem reinicia a máquina. Para desativar o uso do **Ctrl+Alt+Del**, como forma de reiniciar o sistema sem a necessidade de login, devemos editar o arquivo **/etc/inittab**.

Vamos editar o arquivo **/etc/inittab** com um editor de texto, navegamos até a linha que inclui a chamada para **ctrlaltdel** e adicionar a opção **-a** no **shutdown**.

```
root@fusion:~# vi /etc/inittab

# What to do when CTRL-ALT-DEL is pressed.
ca:12345:ctrlaltdel:/sbin/shutdown -t1 -a -r now
```

Nota: No Debian, esta configuração já vem por padrão.

Para permitir que alguns usuários possam desligar o sistema você deverá criar o arquivo **/etc/shutdown.allow** e incluir os nomes de usuários que podem reiniciar o sistema. Na ausência deste arquivo todos os usuários ainda terão poderes de reiniciar a máquina pressionando **CRTL+ALT+DEL**.

Vamos criar o arquivo desta forma:

```
root@fusion:~# cat > /etc/shutdown.allow
root
w3ll
<Pressione → CTRL+D>

root@fusion:~#
```

Nota: Após a edição do arquivo **/etc/inittab** você deverá executar o comando **init q** para que suas alterações entrem em vigor sem a necessidade de um **restart** do sistema.

Vamos validar usando o usuário **w3ll** (este não é o **W3ll** administrador).

```
w3ll@fusion:~$ whoami
w3ll
shutdown: no authorized users logged in.
```

Figura 13 – Reboot não Autorizado

Observação: Isso também evitará que um servidor **Linux** em um mesmo rack que outro servidor com sistema operacional **Windows®** possa ser reiniciado por engano, pois o **CTRL+ALT+DEL** no **Windows®** tem outro contexto, o de apresentar a tela de **login** e não de reiniciar o sistema operacional.

Enquanto alguns promovem o **CTRL+ALT+DEL**, outros não gostam de usá-lo tanto assim.

Vamos aproveitando para configurar o acesso ao arquivo **/etc/inittab** com o menor privilégio possível, pois, este arquivo é responsável por determinar quais processos serão inicializados durante o boot e durante a operação normal do sistema. Alterações indevidas neste arquivo podem comprometer a disponibilidade e a integridade do sistema.

Autor: Wellington Silva



```
root@fusion:~# chmod 640 /etc/inittab
root@fusion:~# chown root.root /etc/inittab
root@fusion:~#
```

3.4 – Magic SysRq Key

O Linux é um sistema muito estável, porém como qualquer outro software, poderá falhar em algum momento, onde nem **CTRL+ALT+DEL** poderá te ajudar. Neste momento você poderá usar um recurso de **Kernel** chamado “**Magic SysRq key**”. Com ele você poderá desligar o sistema sem comprometer o seu **filesystem** (coisa que um desligamento forçado, pressionando o botão de liga/desliga faz).

A opção que permite reiniciar a máquina é **ALT+SysRq+b** e tem o valor decimal **128**. Podemos ver se ele está habilitado observando se existe algum valor na chave **/proc/sys/kernel/sysrq**, qualquer valor diferente de 0 (zero) ele está habilitado. Além de poder ser manipulado por opções de Kernel na **chave /proc/sysrq-trigger**.

```
root@fusion:~# cat /proc/sys/kernel/sysrq
438
root@fusion:~#
```

Então vamos desabilitar:

```
root@fusion:~# cat /proc/sys/kernel/sysrq
438

root@fusion:~# sysctl -w kernel.sysrq=0
kernel.sysrq = 0

root@fusion:~# cat /proc/sys/kernel/sysrq
0
root@fusion:~#
```

Para fixarmos este parâmetro, vamos configure-lo no arquivo **/etc/sysctl.conf** assim:

```
root@fusion:~# vi /etc/sysctl.conf
kernel.sysrq = 0

root@fusion:~# sysctl -p /etc/sysctl.conf
kernel.sysrq = 0

root@fusion:~#
```

Pronto o recurso está desabilitado.

Observação: Aqui estamos radicalizando, mas daria para tirar somente os recursos mais perigosos como, por exemplo, o reboot, subtraindo **128** de **438** e passando o resultado para o **/proc/sys/kernel/sysrq**.

Para um estudo mais aprofundado indico o artigo **001 – Magic SysRq Key** no diretório **Linux/Segurança**.

Autor: Wellington Silva



3.5 – Bloqueando o Terminal

Quando abandonamos a console de um sistema em que estamos trabalhando é importante que ele tenha algum recurso que desconecte automaticamente após um período inativo ou ainda uma forma de bloqueamos a console na necessidade de nos ausentarmos. Para isso podemos usar a variável global **TMOUT** e o programa **vlock** respectivamente.

Vamos configurar a variável **TMOUT** da seguinte forma:

```
root@fusion:~# vi /etc/profile

# /etc/profile: system-wide .profile file for the Bourne shell (sh(1))
# and Bourne compatible shells (bash(1), ksh(1), ash(1), ...).

if [ "`id -u`" -eq 0 ]; then
    PATH="/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin"
else
    PATH="/usr/local/bin:/usr/bin:/bin:/usr/local/games:/usr/games"
fi

readonly TMOUT=300

export PATH TMOUT
"/etc/profile" 40L, 841C written

root@fusion:~#
```

Agora basta recarregar o arquivo **/etc/profile** para entrar em vigor a nova variável:

```
root@fusion:~# source /etc/profile

root@fusion:~# timed out waiting for input: auto-logout
```

Pronto, após 5 minutos (300 segundos) você será "**deslogado**" automaticamente.

Agora vamos instalar o **vlock** para bloquearmos nossa console na necessidade de nos ausentarmos.

```
root@fusion:~# aptitude install vlock
The following NEW packages will be installed:
  vlock
0 packages upgraded, 1 newly installed, 0 to remove and 0 not upgraded.
Need to get 38.1 kB of archives. After unpacking 221 kB will be used.
Get:1 http://ftp.us.debian.org/debian/ squeeze/main vlock i386 2.2.2-3 [38.1 kB]
Fetched 38.1 kB in 6s (6,205 B/s)
Selecting previously deselected package vlock.
(Reading database ... 16351 files and directories currently installed.)
Unpacking vlock (from .../vlock_2.2.2-3_i386.deb) ...
Processing triggers for man-db ...
Setting up vlock (2.2.2-3) ...

root@fusion:~# vlock -a

The entire console display is now completely locked.
You will not be able to switch to another virtual console.
```

Autor: Wellington Silva



```
Please press [ENTER] to unlock
```

Pronto podemos usar agora o comando **vlock** com a opção **-a** (de **all** – todos os terminais) para bloquear a console.

Observação: Ele só irá funcionar no console, nos terminais remotos ele não funciona, já a variável de ambiente funciona, pois ela é carregada no perfil do usuário.

3.6 – Remover Arquivo de Compatibilidade

Antes mesmo de explicarmos como utilizar os módulos do **PAM**, vamos retirar alguns arquivos desnecessários, para deixarmos nosso sistema mais limpo e também evitarmos problemas causados por arquivos de compatibilidade.

Originalmente, o **PAM** usava o **/etc/pam.conf** como seu arquivo de configuração. Mas, hoje em dia, esse arquivo só é necessário se o diretório de configuração padrão do **PAM** não existir. Portanto devemos remover o arquivo **pam.conf**, já que todas as informações de configuração do **PAM** estão em um único diretório, denominado **/etc/pam.d**.

Cada serviço tem seu próprio arquivo de configuração no **/etc/pam.d**, e o nome desse arquivo corresponde ao do programa ou serviço. Por exemplo, o aplicativo de **login** (**/bin/login**) está configurado em **/etc/pam.d/login**. Os programadores definem os nomes dos serviços ou aplicativos.

```
root@fusion:~# cat /etc/pam.conf
# -----#
# /etc/pam.conf                                #
# -----#
#
# NOTE
# ----
#
# NOTE: Most program use a file under the /etc/pam.d/ directory to setup
their
# PAM service modules. This file is used only if that directory does not
exist.
# -----#

# Format:
# serv. module      ctrl      module [path]      ...[args..]      #
# name type        flag                                     #

root@fusion:~# rm /etc/pam.conf
root@fusion:~#
```

3.7 – Restringindo Usuários com Senhas Nulas de Logar na Console

Para testarmos essa funcionalidade vamos usar o usuário **w3ll**, e para isso, vamos retirar o mapeamento da senha dele, no arquivo **/etc/passwd**.

```
root@fusion:~# vi /etc/passwd
root:x:0:0:root:/root:/bin/bash
```

Autor: Wellington Silva




```
--==[ Resumido ]==--
w31l::1000:1000:w31l,,,:/home/w31l:/bin/bash
sshd:x:101:65534::/var/run/sshd:/bin/false
W31l:x:1001:1001::/home/W31l:/bin/bash
~
"/etc/passwd" 22L, 940C written
root@fusion:~#
```

Observe que foi retirado o **x** do usuário **w31l** para que ele não seja questionado quanto à senha.

Vamos testar:

```
Debian GNU/Linux 6.0 fusion tty1
fusion login: w31l
Last login: Tue Jul 31 07:39:08 BRT 2012 on tty1
Linux fusion 2.6.32-5-686 #1 SMP Sun May 6 04:01:19 UTC 2012 i686

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
w31l@fusion:~$ _
```

Figura 14 – Usuário Logando sem a Necessidade de Senha

Observe que não foi solicitada senha para o usuário, isso porque na política padrão do **PAM** de interface **auth**, está permitindo senhas nulas com o parâmetro **nullok_secure**, para o módulo **pam_unix.so**.

Quando utilizamos o **PAM**, podemos configurar o arquivo **/etc/pam.d/login** para desativar a possibilidade de fazer login sem senhas. Esta característica pode ser limitada removendo o argumento **nullok_secure** da seguinte linha do arquivo **/etc/pam.d/common-auth**:

```
root@fusion:~# vi /etc/pam.d/common-auth

# here are the per-package modules (the "Primary" block)
#auth [success=1 default=ignore] pam_unix.so nullok_secure
auth [success=1 default=ignore] pam_unix.so

"/etc/pam.d/common-auth" 26L, 1266C written
root@fusion:~#
```

Assim um usuário com senha nula não poderá autenticar-se no sistema.

3.8 – Bloqueando Usuários Root na Console

Por questões de segurança não é bom que o usuário **root** tenha acesso direto ao sistema (nem local, nem remoto). O ideal é "**logarmos**" como um usuário comum e, quando for necessário fazer alguma tarefa administrativa, usamos o comando **su** para nos tornarmos **root** ou usamos comando **sudo** para executar algum comando como **root**.

Autor: Wellington Silva



Irei explicar duas formas de bloquear o **login** do **root** nos terminais, usando o **PAM** e pelo arquivo **/etc/securetty**.

Vamos tirar a permissão do usuário **root** de acessar diretamente a máquina através do terminal. Para isso, vamos editar o arquivo **/etc/pam.d/login**.

Vamos descomentar a seguinte linha:

```
root@fusion:~# vi /etc/pam.d/login
--==[ Resumido ]==--

# Uncomment and edit /etc/security/time.conf if you need to set
# time restrainst on logins.
# (Replaces the `PORTTIME_CHECKS_ENAB' option from login.defs
# as well as /etc/porttime)
account    requisite pam_time.so

#
```

Agora edite o arquivo **/etc/security/time.conf** adicionando a seguinte linha:

```
# vi /etc/security/time.conf
--==[ Resumido ]==--

# from pseudo terminals at the weekend and on Mondays.
#xsh;ttyp*;root;!WdMo0000-2400

login;*;root;!A10000-2400

#
# End of example file.

root@fusion:~#
```

Onde:

- **login** → Nome do serviço em que a política se aplica;
- ***** → Quais terminais a política se aplica (**tty**), no nosso caso todos terminais;
- **root** → A qual usuário a política se aplica, em nosso caso ao usuário root;
- **A10000-2400** → Em quais dias e horários a política se aplica, em nosso caso todos os dias da semana (**A1**) durante o dia todo (das 0h às 24h);
- **!** → É uma exceção.

No caso o usuário **root** não poderá logar durante o dia todo, todos os dias da semana, no serviço de **login**, o qual é usado para logon local.

A outra forma é através do arquivo **/etc/securetty** de forma que sejam listadas as entradas dos dispositivos **TTY** (listados em **/dev**) que o usuário root e outros usuários privilegiados (uid=0) poderão utilizar para acessar o sistema.

O arquivo **/etc/securetty** é lido pelo programa de **login** (geralmente **/bin/login**) e contém a lista dos dispositivos TTY que o root e outros usuários privilegiados podem utilizar para fazer o login. Para impedir o acesso através de um dispositivo qualquer, basta comentar sua entrada ou remover a linha correspondente no arquivo.

Autor: Wellington Silva



Para impedir o root de fazer login a partir de qualquer dispositivo, basta comentar todas as linhas do arquivo, ou deixá-lo em branco. Adicionalmente, não devemos apagar o arquivo, caso contrário será permitido o login de qualquer dispositivo.

Para nosso exemplo vamos alterar o arquivo **/etc/securetty** deixando-o assim:

```
root@fusion:~# vi /etc/securetty

--==[ Resumido ]==--

# =====
#
# TTYs sorted by major number according to Documentation/devices.txt
#
# =====

# Virtual consoles
#tty1
#tty2
#tty3
#tty4
#tty5
#tty6
tty7
tty8

root@fusion:~#
```

```
Debian GNU/Linux 6.0 fusion tty6
fusion login: root
Password:
Permission denied
Debian GNU/Linux 6.0 fusion tty6
fusion login: _
```

Figura 15 – Login Root Negado

Pronto, com isso o usuário **root** não terá permissão de logar localmente.

Vamos aproveitando para configurar o acesso ao arquivo **/etc/securetty** com o menor privilégio possível, pois, este arquivo é responsável por configurar os terminais que os usuários podem fazer login. Desta forma evitamos que atacantes possam alterar seu conteúdo e assim abrir brechas de segurança.

```
root@fusion:~# chmod 640 /etc/securetty

root@fusion:~# chown root.root /etc/securetty

root@fusion:~#
```

3.9 – Limitando os Usuários que Podem Logar no Console

Aqui iremos ver como restringir o número de usuários que podem fazer **login** no console. Aqui usaremos um arquivo para permitir os usuários de fazer o **login** nos terminais texto. Vamos editar o arquivo **/etc/pam.d/login** e descomentar ou adicionar o módulo **pam_listfile.so** da seguinte maneira:

```
root@fusion:~# vi /etc/pam.d/login

auth      required      pam_listfile.so  item=user  sense=allow
file=/etc/user.allow onerr=fail

root@fusion:~#
```

Agora vamos criar o arquivo **/etc/user.allow** e adicionar os login dos usuários que podem logar direto no console. Lembra o usuário que criamos para administrar nosso servidor? Então vamos criar o arquivo e adicioná-lo.

```
root@fusion:~# cat > /etc/user.allow
w3ll
^C

root@fusion:~#
```

Agora vamos testar se o usuário **w3ll** pode “logar”, e logo após “logaremos” com o usuário **W3ll**.

```
Debian GNU/Linux 6.0 fusion tty1

fusion login: w3ll
Password:

Login incorrect
fusion login: root

Login incorrect
fusion login: W3ll
Password:
Last login: Tue Jul 31 08:39:24 BRT 2012 on tty1
Linux fusion 2.6.32-5-686 #1 SMP Sun May 6 04:01:19 UTC 2012 i686

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
W3ll@fusion:~$ _
```

Figura 16 – Contas Sem Acesso Direto a Console

Pronto, somente os usuários listados no arquivo **/etc/user.allow** poderão fazer o **login** local.

3.10 – Limitar Logins Consecutivos nos Terminais

Para evitarmos que um único usuário seja usado por diversas pessoas, podemos restringir o número de **logins** consecutivos que este usuário pode fazer.

Consulte no arquivo **/etc/pam.d/login** é veja se o módulo **pam_limits.so** está ativado (descomentado).

Autor: Wellington Silva



```

root@fusion:~# vi /etc/pam.d/login

--==[ Resumido ]==--

# Sets up user limits according to /etc/security/limits.conf
# (Replaces the use of /etc/limits in old login)
session required      pam_limits.so

"/etc/pam.d/login" 109L, 4665C written

root@fusion:~#

```

Agora vamos inserir a seguinte linha no arquivo **/etc/security/limits.conf**. Para limitarmos o número de sessões abertas.

```

root@fusion:~# vi /etc/security/limits.conf

--==[ Resumido ]==--

#@student      -          maxlogins       4

*              hard      maxlogins       3

# End of file
"/etc/security/limits.conf" 58L, 2172C written

root@fusion:~#

```

Desta forma, limitamos o usuário a utilizar três terminais consecutivos.

Observação: Iremos usar este recurso mais a frente, para limitarmos alguns serviços, além disso, para testarmos, devemos liberar mais um terminal no arquivo **/etc/inittab**.

Aproveitando que estamos configurando o arquivo de limites, vamos acrescentar mais esta linha:

```

root@fusion:~# vi /etc/security/limits.conf

--==[ Resumido ]==--

#@student      -          maxlogins       4

*              hard      maxlogins       3
*              hard      nproc          100

# End of file
"/etc/security/limits.conf" 58L, 2172C written

root@fusion:~#

```

Desta forma nós inibimos o comando da morte (**fork bomb**), que pode ser feito por qualquer usuário (com poder de apenas "logar" localmente e tendo uma shell válida) de causar um **DoS (Deny of Service)** no sistema. Experimente logar com um usuário simples e executar o seguinte comando:

```
$ :(){ :|:& };;
```

Seu terminal travará, com o uso do **nproc** podemos limitar o número máximo de processos por usuário ou para todos os usuários, que foi o nosso caso.

Autor: Wellington Silva



3.11 – Usuários que Podem Usar os Comandos su e sudo

Para evitar que usuários que não tenha permissão de uso dos comandos **su/sudo** possam elevar seus privilégios para outros usuários do sistema, vamos criar um grupo e adicionar somente usuários que poderão usar o comando **su** e outro para usuários que possam usar o **sudo**:

```
root@fusion:~# groupadd pwrgrp
root@fusion:~# groupadd supgrp
root@fusion:~# usermod -G pwrgrp w3ll
root@fusion:~# usermod -G supgrp W3ll
root@fusion:~#
```

Criamos o grupo **pwrgrp (Power Group)** para uma administração limitada (iremos ver as limitações deste grupo mais a frente) e outro grupo **supgrp (Super Group)** para acesso administrativo.

Agora vamos configurar os arquivos **/etc/pam.d/su** e **/etc/pam.d/sudo** para criarmos as políticas de uso dos respectivos comandos **su** e **sudo**.

```
root@fusion:~# vi /etc/pam.d/su
--==[ Resumido ]==--
auth      required      pam_wheel.so    group=supgrp

# Uncomment this if you want members of a specific group to not
# be allowed to use su at all.
"/etc/pam.d/su" 63L, 2345C written

# vi /etc/pam.d/sudo
##PAM-1.0
auth      required      pam_wheel.so    group=pwrgrp
@include common-auth
@include common-account

session required pam_permit.so
session required pam_limits.so

"/etc/pam.d/sudo" 8L, 159C written
root@fusion:~#
```

Desta maneira, somente usuários que pertencerem ao grupo **supgrp** poderá usar o comando **su** e somente usuários pertencentes ao grupo **pwrgrp** poderá usar o comando **sudo**.

Observação: O grupo usado não poderá ser o grupo primário dos usuários.

Os usuários que podem elevar seus privilégios com a utilização do comando **su**, devem ter seus acessos registrados, para fins de auditoria e análise forense em caso de abusos.

Autor: Wellington Silva



Para isso, vamos agora editar o arquivo **/etc/login.defs** descomentando a seguinte linha:

```
root@fusion:~# vi /etc/login.defs

-==[ Resumido ]==--

#
# If defined, all su activity is logged to this file.
#
SULOG_FILE          /var/log/sulog

"/etc/login.defs" 335L, 10182C written
root@fusion:~#
```

Feito, agora é só continuarmos o ciclo de segurança no console.

Nota: Em sistemas **RedHat-like**, os logs do comando **su** são auditados através do arquivo **/var/log/secure**.

4 – Mensagens de Advertência ao Logar no Sistema

A exibição de uma mensagem de advertência durante o Login tem como propósito deixar claro que o sistema em questão pertence à empresa, sendo de uso restrito para usuários autorizados e que os acessos serão gravados em log. Do ponto de vista legal, é recomendável deixar claro que o uso do sistema ou serviço é restrito e que o acesso indevido poderá ter consequências.

Para isso, podemos editar os arquivos **/etc/issue** e **/etc/issue.net** para ser mostrada a mensagem assim que o usuário entrar no sistema localmente ou remoto, veja o exemplo de mensagem:

```
root@fusion:~# cat /etc/issue
***** WELCOME! *****
WARNING NOTICE: This is a private system for use by MyCompany the unauthorized
access or attempt, use or modification of this system is strictly prohibited.
Individuals undertaking such unauthorized access, use or modification are
subject to company disciplinary proceedings and/or criminal and civil
penalties under applicable domestic and foreign laws. The use of this system
may be monitored and recorded for administrative and security reasons in
accordance with local law.

***** BEM VINDO! *****
AVISO: Este e um sistema privado para uso exclusivo da MyCompany. Acesso não
autorizados ou tentativas, uso ou modificacao deste sistema sao estritamente
restritos. Indivíduos que executam tal acesso, usam ou modificam sem
autorização estão sujeitos a procedimentos disciplinares e penalidade civil
com base em leis domestica e estrangeiras aplicáveis. O uso deste sistema
pode ser monitorado e armazenado para uso posterior conforme a lei local.

root@fusion:~#
```

Autor: Wellington Silva



5 – Referencias Bibliográficas

[1] Morimoto, Carlos. Disponível em: <<http://www.hardware.com.br/termos/bios>>. Acessado em: 21/07/2012.

[2] Duarte, Helto, <<http://heltonduarte.com/2009/07/03/gerenciadores-de-boot/>> Acessado em 21/07/2012.

[3] Ribeiro, Uira – Certificação Linux, 1ª Ed, São Paulo, 2004, Axcel Books

[4] Manual do GNU GRUB v2. Disponível em: <http://www.gnu.org/software/grub/manual/html_node/Security.html#Security>. Acessado em: 24/07/2012.

[5] Drs305. Disponível em: <<http://ubuntuforums.org/showthread.php?t=1369019>>. Acessado em: 24/07/2012.

[6] Artigo. Disponível em: <<http://www.linuxhowtos.org/Tips%20and%20Tricks/sysrq.htm>> . Acessado em : 28/07/2012.

[7] Manual Debian. Disponível em: <http://www.debian.org/doc/manuals/debian-reference/ch09.pt.html#_alt_sysrq_key>. Acessado em: 31/07/2012.

[8] Documentação Kernel. Disponível em: <<http://Kernel.org/doc/Documentation/sysrq.txt>>. Acessado em: 31/07/2012.

[9] Debian Administrator. Disponível em: <http://www.debian-administration.org/article/457/The_magic_sysrq_options_introduced> . Acessado em: 28/07/2012.

[10] Terpstra, John; Love, Paul; Reck, Ronald; Scanlon, Tim – Segurança para Linux, 1ª Ed, 2005,